

ANÁLISIS DE CASO

GIMNASIO

Diego Iván Méndez López

Alfonso Canseco Bustos

Gerald Díaz Venegas del Castillo

1. Preparación: Importación de Librerías

El primer paso en cualquier proyecto de análisis de datos es configurar nuestro entorno. Aquí, importamos las herramientas esenciales de Python que nos permitirán manipular, visualizar y modelar los datos.

```
import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
import seaborn as sns  
import plotly.express as px  
import plotly.figure_factory as ff  
from sklearn.preprocessing import StandardScaler  
from sklearn.cluster import KMeans  
from sklearn.metrics import silhouette_score  
from sklearn.decomposition import PCA
```

- **pandas y numpy:** Para manejo y manipulación eficiente de datos tabulares.
- **matplotlib, seaborn y plotly:** Para crear visualizaciones estáticas y dinámicas.
- **sklearn:** La biblioteca clave para tareas de Machine Learning, incluyendo escalado de datos, algoritmos de clustering y métricas de evaluación.

2. Carga y Exploración Inicial de Datos

Una vez cargadas las herramientas, procedemos a introducir nuestros datos y realizar una primera inspección. Esto nos da una visión rápida de la estructura del dataset, identificando posibles problemas como valores faltantes o tipos de datos incorrectos.

```
try:  
    df = pd.read_csv("escenario_gimnasios_15.csv", encoding="utf-8-sig")  
except FileNotFoundError:  
    print("
```

```
>>> %Run retoprueba.py  
X No se encontró el archivo CSV  
  
Process ended with exit code 0.
```

2. Definir propiedades tabla

```
print("Forma (filas, columnas):", df.shape)
print("\nTipos de datos:\n", df.dtypes)
print("\nValores faltantes por columna:\n", df.isna().sum())

# Mostrar todas las columnas
pd.set_option('display.max_columns', None)
pd.set_option('display.width', 1000)

# Estadística básica
desc = df.describe(include="all")
print("\nDescripción numérica:\n", desc)

# Variables clave
var1 = "Ventas_Mensuales"
var2 = "Precio_Promedio"
```

Descripción general:																		
	Gimnasio	Zona	Metros_Cuadrados	Entrenadores	Trafico_Mensual	Precio_Membresia	Descuento_Promedio	Gasto_Marketing	Satisfaccion	Equipos	Churn	Tasa_Conversion	Clases_Semana	Ingresos_Mensuales				
count	15	15	15.000000	15.000000	14.000000	14.000000	14.000000	14.000000	14.000000	15.000000	15.000000	15.000000	15.000000	15.000000	1.500000e+01			
unique	15	5		NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN			
top	Gimnasio_01	Norte		NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN			
freq	1	6		NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN			
mean	NaN	NaN	1443.866667	18.000000	24569.928571	1010.340000	0.088571	29501.071429	7.678571	351.133333	0.071067	0.120867	55.400000	2.508981e+06				
std	NaN	NaN	676.105958	8.510498	9661.150747	299.542216	0.092475	19214.531786	0.786598	121.547091	0.042111	0.042932	32.048624	1.478137e+06				
min	NaN	NaN	528.000000	4.000000	12177.000000	476.820000	0.010000	4803.000000	6.200000	158.000000	0.019000	0.055000	13.000000	3.863880e+05				
25%	NaN	NaN	843.000000	11.000000	17593.000000	747.027500	0.035000	11747.500000	7.275000	280.500000	0.041500	0.087500	31.000000	1.522572e+06				
50%	NaN	NaN	1221.000000	17.000000	20785.000000	1085.190000	0.050000	29504.500000	7.850000	329.000000	0.059000	0.124000	53.000000	2.225298e+06				
75%	NaN	NaN	2035.500000	25.000000	35341.250000	1135.090000	0.080000	46845.250000	8.275000	441.000000	0.102500	0.154500	76.500000	3.003174e+06				
max	NaN	NaN	2475.000000	31.000000	37512.000000	1489.330000	0.300000	56525.000000	8.800000	588.000000	0.145000	0.180000	115.000000	6.270945e+06				

3. Medidas de Tendencia Central

Para comprender mejor la distribución de nuestros datos, calculamos las medidas de tendencia central: **media, mediana y moda**. Estas estadísticas nos dan una idea de los valores "típicos" dentro de nuestras variables clave.

```
def mostrar_medidas(df, variables):
    for var in variables:
        media = df[var].mean(skipna=True)
        mediana = df[var].median(skipna=True)
        moda = df[var].mode(dropna=True)
        moda = moda.iloc[0] if not moda.empty else None
        print(f"\n{var}: Media={media:.2f}, Mediana={mediana:.2f},
Moda={moda}")

variables_clave = ["Ingresos_Mensuales", "Precio_Membresia"]
mostrar_medidas(df, variables_clave)
```

- **Media:** El promedio aritmético, sensible a valores atípicos.
- **Mediana:** El valor central, robusto frente a valores atípicos.
- **Moda:** El valor que más se repite, útil para datos categóricos o discretos.

Al aplicar estas medidas a "Ingresos_Mensuales" y "Precio_Membresia", obtenemos una instantánea de los patrones de comportamiento económico de nuestros clientes.

```
🔗 Medidas de tendencia central
-----
Ingresos_Mensuales:
    Media      = 2508981.00
    Mediana   = 2225298.00
    Moda       = 386388
-----
Precio_Membresia:
    Media      = 1010.34
    Mediana   = 1085.19
    Moda       = 476.82
```

4. Visualización de Datos: Estáticos e Interactivos

Gráficos Estáticos (PNG)

```
def graficos_estaticos(df, var1, var2, corr):
    # Boxplot Ingresos
    plt.figure(figsize=(6, 4))
    df[var1].plot(kind="box", title=f"Boxplot - {var1}")
    plt.tight_layout()
    plt.savefig("boxplot_ingresos.png", dpi=120)
    plt.close()

    # Histograma Ingresos
    plt.figure(figsize=(6, 4))
    df[var1].plot(kind="hist", bins=10, title=f"Histograma - {var1}")
    plt.xlabel("Valor")
    plt.ylabel("Frecuencia")
    plt.tight_layout()
    plt.savefig("hist_ingresos.png", dpi=120)
    plt.close()

    # Boxplot Precio Membresía
    plt.figure(figsize=(6, 4))
    df[var2].plot(kind="box", title=f"Boxplot - {var2}")
    plt.tight_layout()
    plt.savefig("boxplot_precio_membresia.png", dpi=120)
    plt.close()

    # Heatmap de correlaciones
    plt.figure(figsize=(8, 6))
    sns.set(font_scale=0.8)
    sns.heatmap(corr, annot=True, fmt=".2f", square=True,
                annot_kws={"size": 7}, cmap="coolwarm",
                cbar_kws={"shrink": 0.8})
    plt.title("Heatmap de correlaciones (variables numéricas)", fontsize=10)
    plt.xticks(rotation=45, ha="right", fontsize=8)
    plt.yticks(fontsize=8)
    plt.tight_layout()
    plt.savefig("heatmap_correlaciones.png", dpi=120)
    plt.close()

    print("\n Gráficos estáticos guardados como imágenes PNG.")
```

Gráficos Interactivos (Plotly)

```
def graficos_interactivos(df, var1, var2, corr):
    # Boxplot interactivo
    fig_box = px.box(df, y=var1, title=f"Boxplot interactivo - {var1}")
    fig_box.show()

    # Histograma interactivo
    fig_hist = px.histogram(df, x=var1, nbins=15,
                           title=f"Histograma interactivo - {var1}")
    fig_hist.show()

    # Scatter interactivo
    fig_scatter = px.scatter(df, x=var2, y=var1,
                             size=var1, color=var2,
                             hover_data=df.columns,
                             title=f"Relación entre {var1} y {var2}")
    fig_scatter.show()

    # Heatmap interactivo
    fig_heatmap = ff.create_annotated_heatmap(
        z=corr.values,
        x=list(corr.columns),
        y=list(corr.index),
        annotation_text=corr.round(2).values,
        colorscale="RdBu",
        showscale=True
    )
    fig_heatmap.update_layout(title="Heatmap interactivo de correlaciones")
    fig_heatmap.show()

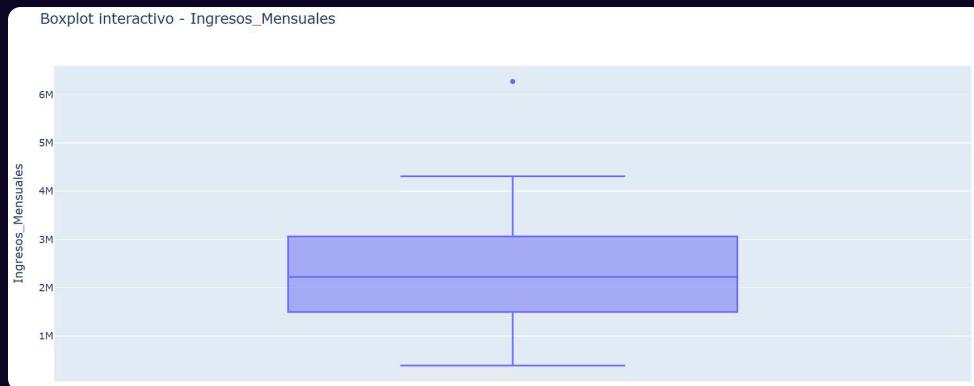
    print("\n Gráficos interactivos desplegados en el navegador/ventana.")
```

Imágenes de las gráficas

1

Boxplot

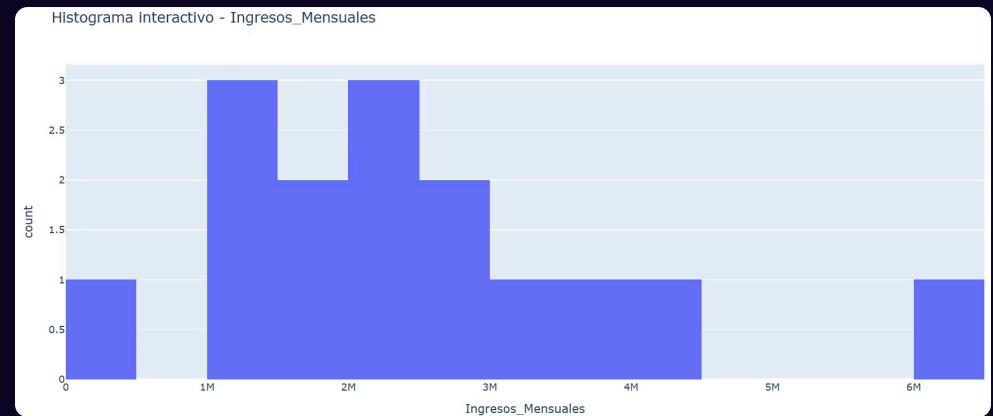
Identifican la distribución de los datos y la presencia de outliers.



2

Histograma

Muestran la frecuencia de los valores en rangos específicos



3

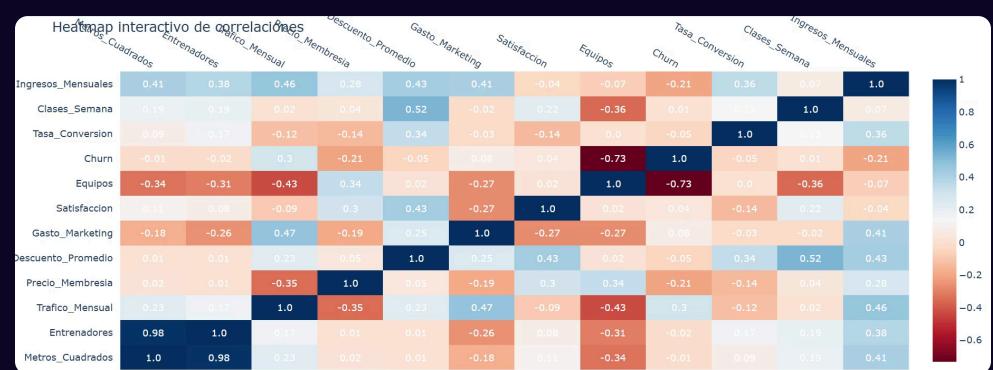
Relacion



4

Heatmap

Visualizan la fuerza y dirección de las relaciones lineales entre variables.



5. Preparación de Datos para Clustering

Antes de aplicar K-Means, es crucial preparar los datos adecuadamente. Esto incluye manejar valores nulos, eliminar variables redundantes y escalar las características.

```
num = df.select_dtypes("number").copy()
num = num.fillna(num.mean())

# Eliminar variables muy correlacionadas
high_corr_threshold = 0.90
corr_matrix = num.corr()
to_drop = set()
for i, c1 in enumerate(corr_matrix.columns):
    for c2 in corr_matrix.columns[i+1:]:
        if abs(corr_matrix.loc[c1, c2]) >= high_corr_threshold:
            to_drop.add(c2)

X = num.drop(columns=list(to_drop)) if to_drop else num.copy()
```

```
# Escalado
scaler = StandardScaler()
Xs = scaler.fit_transform(X)
```

```
==== Selección de variables para clustering ====
Variables usadas: ['Metros_Cuadrados', 'Trafico_Mensual', 'Precio_Membresia', 'Descuento_Promedio', 'Gasto_Marketing', 'Satisfaccion', 'Equipos', 'Churn', 'Tasa_Conversion', 'Clases_Semana', 'Ingresos_Mensuales']
Eliminadas por alta correlación (|r|>=0.90): ['Entrenadores']
```

6. Selección del Número Óptimo de Clusters (K)

Sugerencia automática de k (por máxima silueta en 2.6): k=3

```
k_values = range(2, 7)
inertias = []
sil_scores = []
for k in k_values:
    km = KMeans(n_clusters=k, random_state=42, n_init=10)
    labels = km.fit_predict(Xs)
    inertias.append(km.inertia_)
    sil = silhouette_score(Xs, labels)
    sil_scores.append(sil)

best_k = int(k_values[np.argmax(sil_scores)])
print(f"\nSugerencia automática de k: k={best_k}")
```

En este caso, se sugiere automáticamente el valor de K que maximiza el coeficiente de silueta, indicando una mejor estructura de clúster.

Valor de K es 3.

7. Entrenamiento del Modelo y Análisis de Clusters

```
kmeans = KMeans(n_clusters=best_k, random_state=42, n_init=10)
df["Cluster"] = kmeans.fit_predict(Xs)
```

```
centers_std = kmeans.cluster_centers_
centers_orig = scaler.inverse_transform(centers_std)
centers_df = pd.DataFrame(centers_orig, columns=X.columns)
centers_df.index = [f"Cluster_{i}" for i in range(best_k)]
print("\n Centros de K-Means (en escala original):")
print(centers_df.round(2))
```

```
Centros de K-Means (en escala original):
   Metros_Cuadrados  Trafico_Mensual  Precio_Membresia  Descuento_Promedio  Gasto_Marketing  Satisfaccion  Equipos  Churn  Tasa_Conversion  Clases_Semana  Ingresos_Mensuales
Cluster_0          1713.75        36373.00         735.02           0.06          41424.00       7.22    248.75     0.10      0.10          48.75        2722544.75
Cluster_1          1297.00        18277.55        1111.70           0.06          21270.67       7.70    400.89     0.06      0.12          48.00        2035250.78
Cluster_2          1565.00        29279.50        1104.84           0.30          42692.04       8.50    332.00     0.06      0.15         102.00        4213639.50
```

```
Medias por cluster:
   Ingresos_Mensuales  Precio_Membresia
Cluster
0             2.722545e+06      643.253333
1             2.035251e+06      1111.703333
2             4.213640e+06      1104.835000
```

1

Asignación de Clusters

Cada fila del DataFrame original ahora tiene una etiqueta de cluster asignada.

2

Centros de los Clusters

Desescalamos los centroides para interpretarlos en la escala original de los datos. Estos nos definen el "cliente tipo" de cada segmento.

3

Conteo de Elementos por Cluster

Verificamos el tamaño de cada cluster para asegurar una distribución razonable.

4

Promedios por Cluster

Calculamos las medias de las variables clave (Ingresos_Mensuales, Precio_Membresia) para cada cluster, lo que nos permite perfilar cada segmento.

8. Selección de gráficos

El usuario sera capaz de elegir la manera en la que quiere los graficos. Ya sea que si los quiera o no.

```
print("1 - Solo gráficos estáticos (PNG)")  
print("2 - Solo gráficos interactivos (Plotly)")  
print("3 - Ambos")  
  
opcion = input("Elige una opción: ")  
  
if opcion == "1":  
    graficos_estaticos(df, var1, var2, corr)  
elif opcion == "2":  
    graficos_interactivos(df, var1, var2, corr)  
elif opcion == "3":  
    graficos_estaticos(df, var1, var2, corr)  
    graficos_interactivos(df, var1, var2, corr)  
  
print("\n Análisis completado.")
```

💡 Selecciona qué tipo de gráficos quieres ver:
1 - Solo gráficos estáticos (PNG)
2 - Solo gráficos interactivos (Plotly)
3 - Ambos
Elige una opción (1, 2 o 3): 3

Conclusiones

El código permite realizar un análisis completo de los datos del gimnasio, desde la exploración inicial hasta la segmentación con K-Means. Genera estadísticas, gráficos estáticos e interactivos, identifica correlaciones y agrupa los gimnasios en clusters, facilitando la interpretación y el diseño de estrategias de negocio personalizadas.

